Interfacing Servo Motor with AVR (Firebird V)

Joel M. Pinto and Vishal H. Rajai
eYantra Summer Internship'14
Embedded Real-Time Systems Lab
Indian Institute of Technology, Bombay

IIT Bombay
July 21, 2014

Agenda of Discussion

1. Introduction
2. Servo motor
   - Principle and Working
   - Operating Servo Motor
   - Selection of Servo motor
3. AVR ATmega2560
   - Generating PWM signals
   - Code

Prerequisite knowledge

1. Basic IO Interfacing using ports
2. Working with Timers and basic knowledge of registers of ATmega2560.

Introduction

- Servo motors (or servos) are self-contained devices that rotate or push parts of a machine with great precision.
- Servos can put out about 42 oz/in of torque.
- Relatively inexpensive.
- Widely used for educational purpose in mechatronics as they can be controlled by a microcontroller.

Principle and Working

- A servo mainly contains:

Principle and Working

- A servo mainly contains:
  - dc motor

Principle and Working

- A servo mainly contains:
  - dc motor
  - gear train

Principle and Working

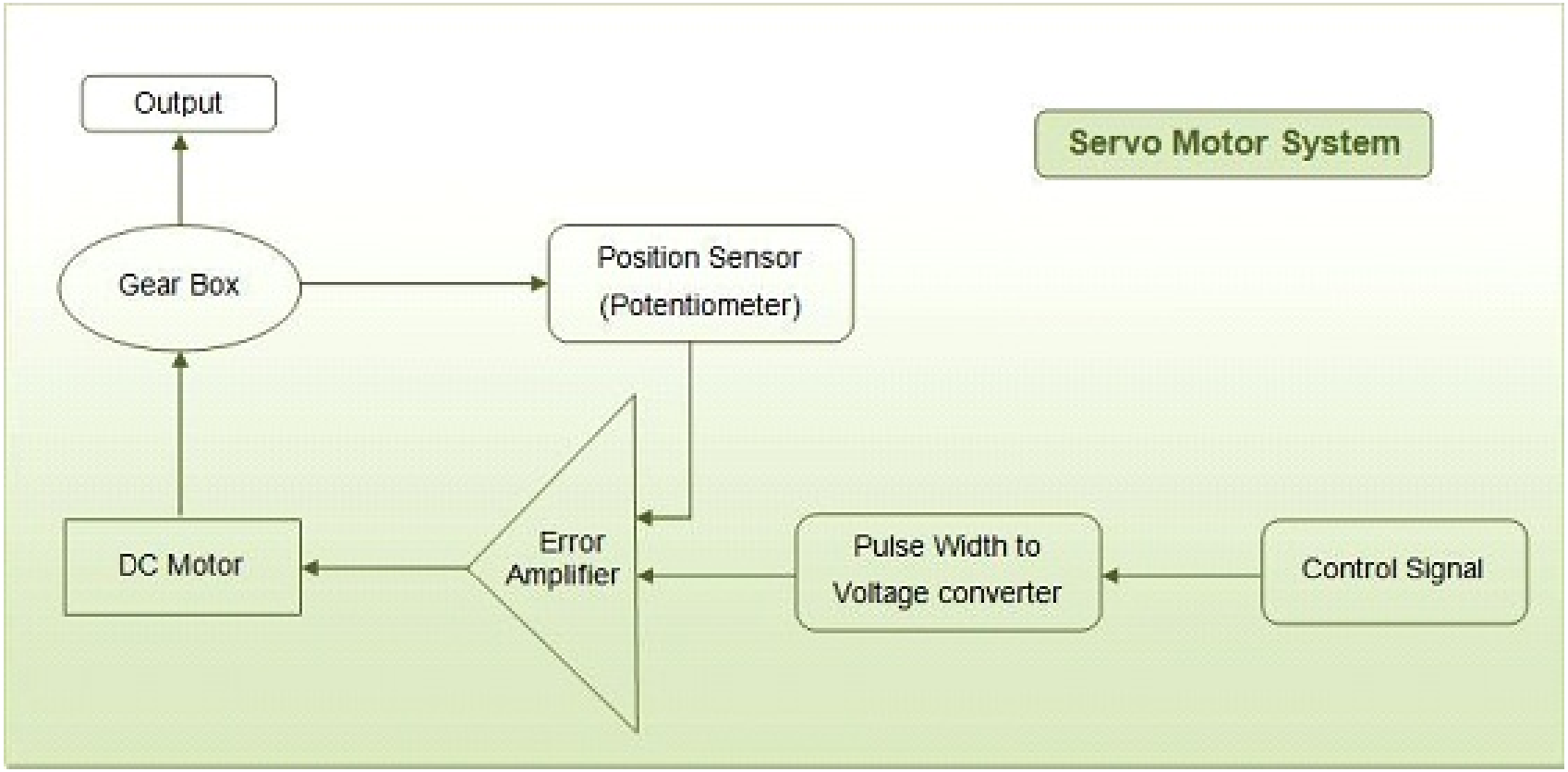
- A servo mainly contains:
  - dc motor
  - gear train
  - potentiometer

Principle and Working

- A servo mainly contains:
  - dc motor
  - gear train
  - potentiometer
  - control circuitry.

Principle and Working

- A servo mainly contains:
  - dc motor
  - gear train
  - potentiometer
  - control circuitry.
- Forming closed loop control system.


Servo Motor System
Output
Gear Box
Position Sensor
(Potentiometer)
DC Motor
Error
Amplifier
Pulse Width to
Voltage converter
Control Signal

- As shown in figure, control signal and output signal are fed to error detector.

- As shown in figure, control signal and output signal are fed to error detector.
- Resultant signal from error detector acts as input to the dc motor to rotate.

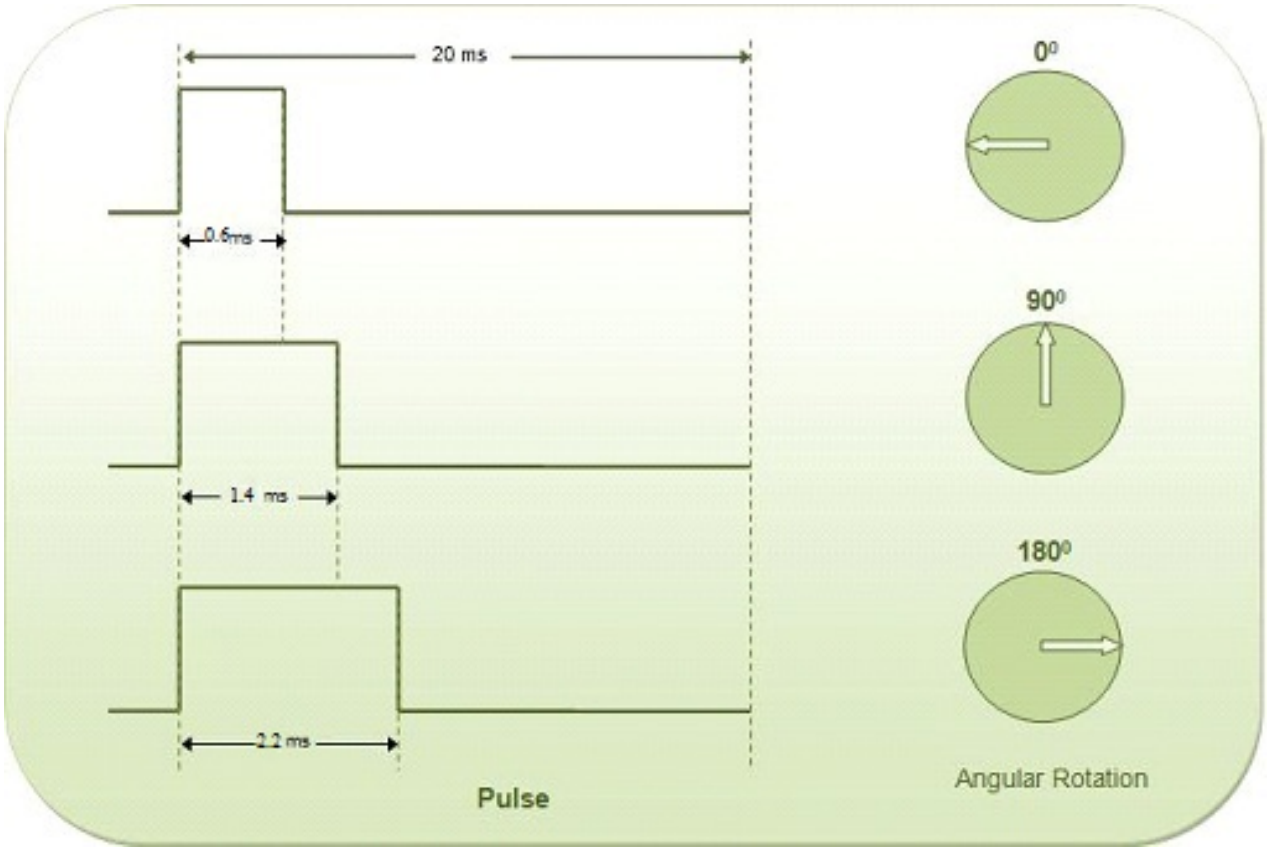
- As shown in figure, control signal and output signal are fed to error detector.
- Resultant signal from error detector acts as input to the dc motor to rotate.
- Also rotating the potentiometer knob coupled with its shaft via gears.

- As shown in figure, control signal and output signal are fed to error detector.
- Resultant signal from error detector acts as input to the dc motor to rotate.
- Also rotating the potentiometer knob coupled with its shaft via gears.
- Reaching desired angle, there would not be any difference in the signals fed to error detector.

- As shown in figure, control signal and output signal are fed to error detector.
- Resultant signal from error detector acts as input to the dc motor to rotate.
- Also rotating the potentiometer knob coupled with its shaft via gears.
- Reaching desired angle, there would not be any difference in the signals fed to error detector.
- Resulting in motor to stop rotating and wait at that position

Operating servo motor

- 'on-time' of a PWM signal is used as control signal to rotate motor at particular angle.



- This time period depends on the servo used and not on total time period or duty cycle of PWM signal.

- This time period depends on the servo used and not on total time period or duty cycle of PWM signal.
- Values of on-time for rotating servo to 0 and 180 degree are provided by the manufacturer in data sheet.

- This time period depends on the servo used and not on total time period or duty cycle of PWM signal.
- Values of on-time for rotating servo to 0 and 180 degree are provided by the manufacturer in data sheet.
- Graph of on-time period vs. angle is linear.

- This time period depends on the servo used and not on total time period or duty cycle of PWM signal.
- Values of on-time for rotating servo to 0 and 180 degree are provided by the manufacturer in data sheet.
- Graph of on-time period vs. angle is linear.
- Range of PWM frequency for operating servo is 40-60 Hz

Selection of Servo motor

Servo is selected based on its:

Selection of Servo motor

Servo is selected based on its:

- Torque

Selection of Servo motor

Servo is selected based on its:

- Torque
- Speed

Selection of Servo motor

Servo is selected based on its:

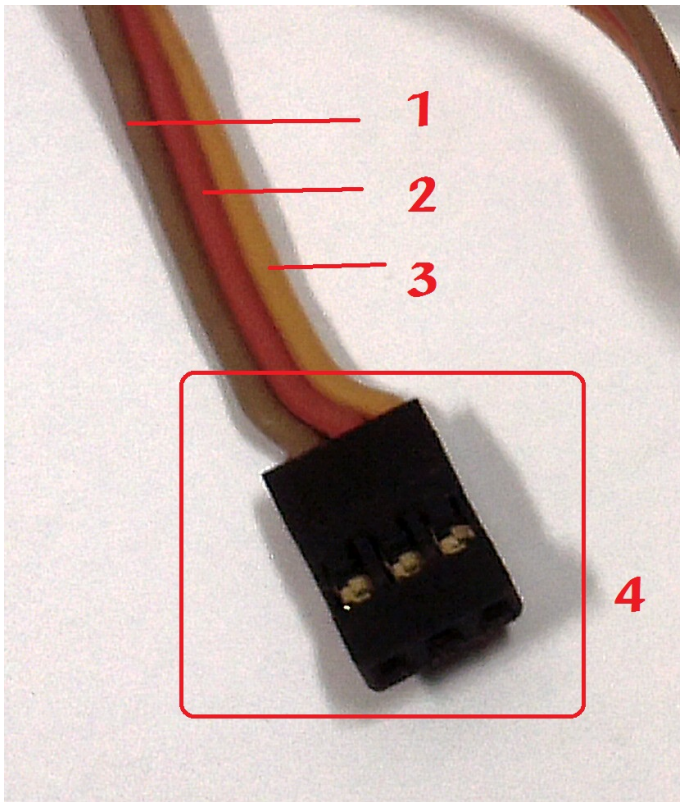
- Torque
- Speed
- Weight

Selection of Servo motor

Servo is selected based on its:

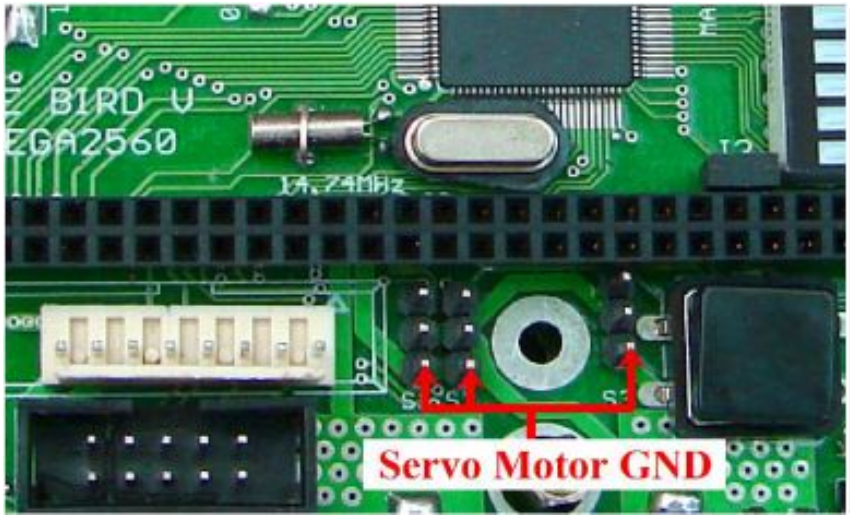
- Torque
- Speed
- Weight
- Dimensions

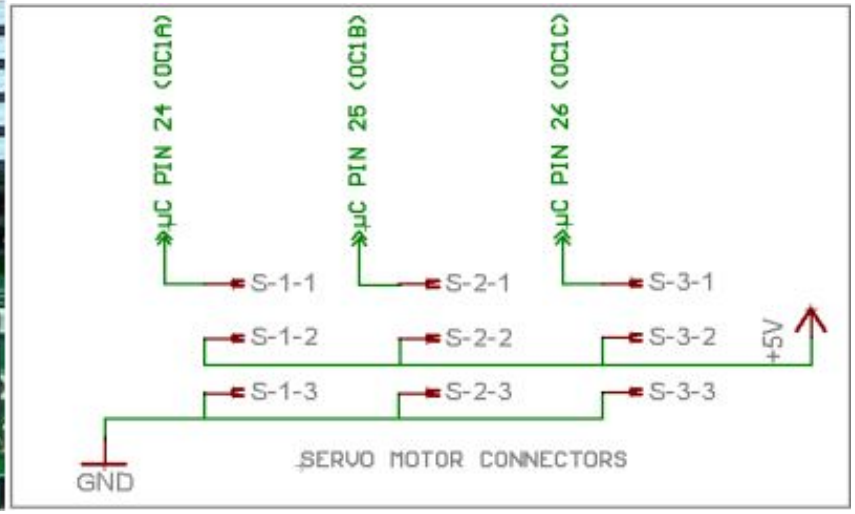
Servo connector



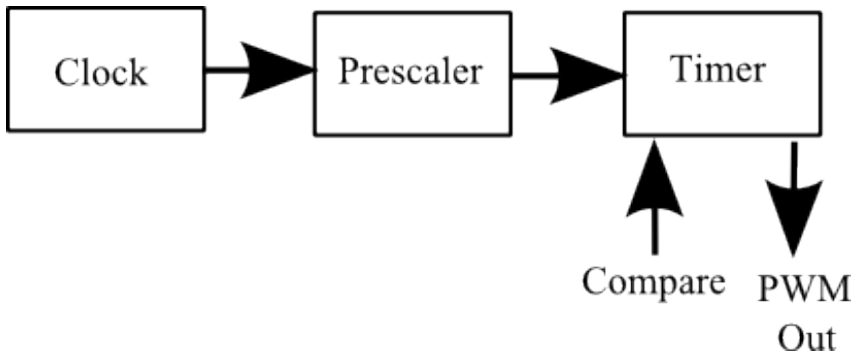
- 1– Ground
- 2– Power
- 3– Control signal
- 4– Connector to controller

Interfacing servo with Firebird V





Using Timer1 for PWM generation



- Timer1 in Fast PWM mode is used here.
- Fast PWM in mode 14 is used to rotate servo motor.

11	1	0	1	1	PWM, Phase Correct	OCRnA	TOP	BOTTOM
12	1	1	0	0	CTC	ICRn	Immediate	MAX
13	1	1	0	1	(Reserved)	–	–	–
14	1	1	1	0	Fast PWM	ICRn	BOTTOM	TOP
15	1	1	1	1	Fast PWM	OCRnA	BOTTOM	TOP

WGM bit description

Generating PWM signals contd.

- Counter counts from BOTTOM=0 to TOP

Generating PWM signals contd.

- Counter counts from BOTTOM=0 to TOP
- And sets the output as high till compare value is greater than counter value

Generating PWM signals contd.

- Counter counts from BOTTOM=0 to TOP
- And sets the output as high till compare value is greater than counter value
- Clears the output when the count value matches the value in compare register till the next cycle starts.

Generating PWM signals contd.

- Counter counts from BOTTOM=0 to TOP
- And sets the output as high till compare value is greater than counter value
- Clears the output when the count value matches the value in compare register till the next cycle starts.
- Let the Prescaler, N=256

Generating PWM signals contd.

- Counter counts from BOTTOM=0 to TOP
- And sets the output as high till compare value is greater than counter value
- Clears the output when the count value matches the value in compare register till the next cycle starts.
- Let the Prescaler, N=256
- $f_{cpu} = 14745600Hz$

Generating PWM signals contd.

- TOP is calculated from desired PWM frequency as:

Generating PWM signals contd.

- TOP is calculated from desired PWM frequency as:

$$f_{PWM} = \frac{f_{cpu}}{N * (1 + TOP)}$$

So, $TOP = \frac{f_{cpu}}{N * f_{PWM}} - 1 = 1023$

Generating PWM signals contd.

- TOP is calculated from desired PWM frequency as:

  $$f_{PWM} = \frac{f_{cpu}}{N * (1 + TOP)}$$

  So, $TOP = \frac{f_{cpu}}{N * f_{PWM}} - 1 = 1023$
- ICR1=TOP
- Value of channel A of OCR register will be:

Generating PWM signals contd.

- TOP is calculated from desired PWM frequency as:
  $f_{PWM} = \frac{f_{cpu}}{N * (1 + TOP)}$
  So, $TOP = \frac{f_{cpu}}{N * f_{PWM}} - 1 = 1023$
- ICR1=TOP
- Value of channel A of OCR register will be:
  $OCR1A = T_{on}(\frac{f_{cpu}}{N}) = T_{on} * 57600$

Generating PWM signals contd.

- TOP is calculated from desired PWM frequency as:
  $f_{PWM} = \frac{f_{cpu}}{N * (1 + TOP)}$
  So, $TOP = \frac{f_{cpu}}{N * f_{PWM}} - 1 = 1023$
- ICR1=TOP
- Value of channel A of OCR register will be:
  $OCR1A = T_{on}(\frac{f_{cpu}}{N}) = T_{on} * 57600$
- Values of registers like TCCR1A/B and TCNT1 can be found using ATmega2560 datasheet.

Code
Simple code

Header files

Code
Simple code

Header files

```
#define F_CPU 14745600
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
```

Code
Simple code

Header files

```
#define F_CPU 14745600
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
```

Port Initialization

Code
Simple code

Header files

```
#define F_CPU 14745600
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
```

Port Initialization

```
void port_init()
{
DDRB = DDRB | 0x20;
PORTB = PORTB | 0x20;
}
```

Convert the degrees in terms of counts for Timer1
As the relation between on-time period and corresponding degree is linear, so is the relation between the count value in OCR and degree.

Convert the degrees in terms of counts for Timer1
As the relation between on-time period and corresponding degree is linear, so is the relation between the count value in OCR and degree.

```
float regval = ((float)degrees * 0.512) + 34.56;
OCR1A = (uint16_t) regval;
```

Thank You!

Post your queries on: helpdesk@e-yantra.org